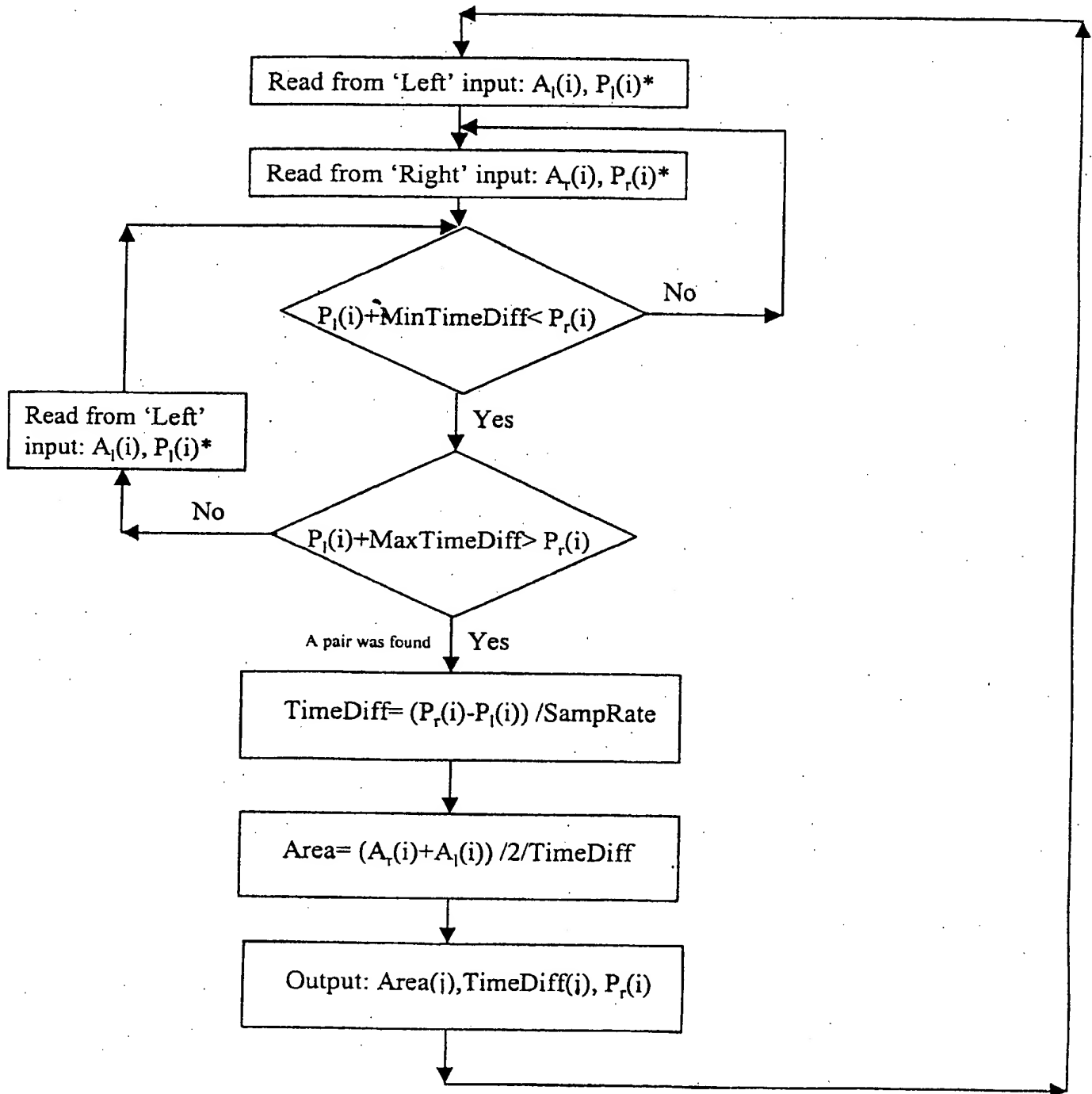<u>ArVlAnalyzer</u>

Input: two files (one for each line scan).

Each file contain 2 vectors one of Positions (P(i)) and the other has the corresponding Area (A(i))

Output: three vectors - Area, TimeDiff (inversely proportional to velocity), Position

Parameters that can be determined - MinTimeDiff, MaxTimeDiff

Read from 'Left' input: $A_l(i)$, $P_l(i)$*

Read from 'Right' input: $A_r(i)$, $P_r(i)$*

$P_l(i)+MinTimeDiff < P_r(i)$ — No

Read from 'Left' input: $A_l(i)$, $P_l(i)$*

Yes

$P_l(i)+MaxTimeDiff > P_r(i)$ — No

A pair was found — Yes

TimeDiff= $(P_r(i)-P_l(i))$ /SampRate

Area= $(A_r(i)+A_l(i))$ /2/TimeDiff

Output: Area(i),TimeDiff(i), $P_r(i)$

Position is presented in point number and not time

TimeDiff is in Seconds and is inversely proportional to the velocity

* Program ends when one of the input files ends

FIG. 8

REPLACEMENT SHEET

ChDiv

Input - two vectors: Y(i) - channel 1 - square wave - chopping signal, 0<=Yi<=1 ⎤ Usually Sampled
                     X(i) - channel 2 - fluorescence raw data - from the detecting region (both line scan) ⎦ at 40KHz
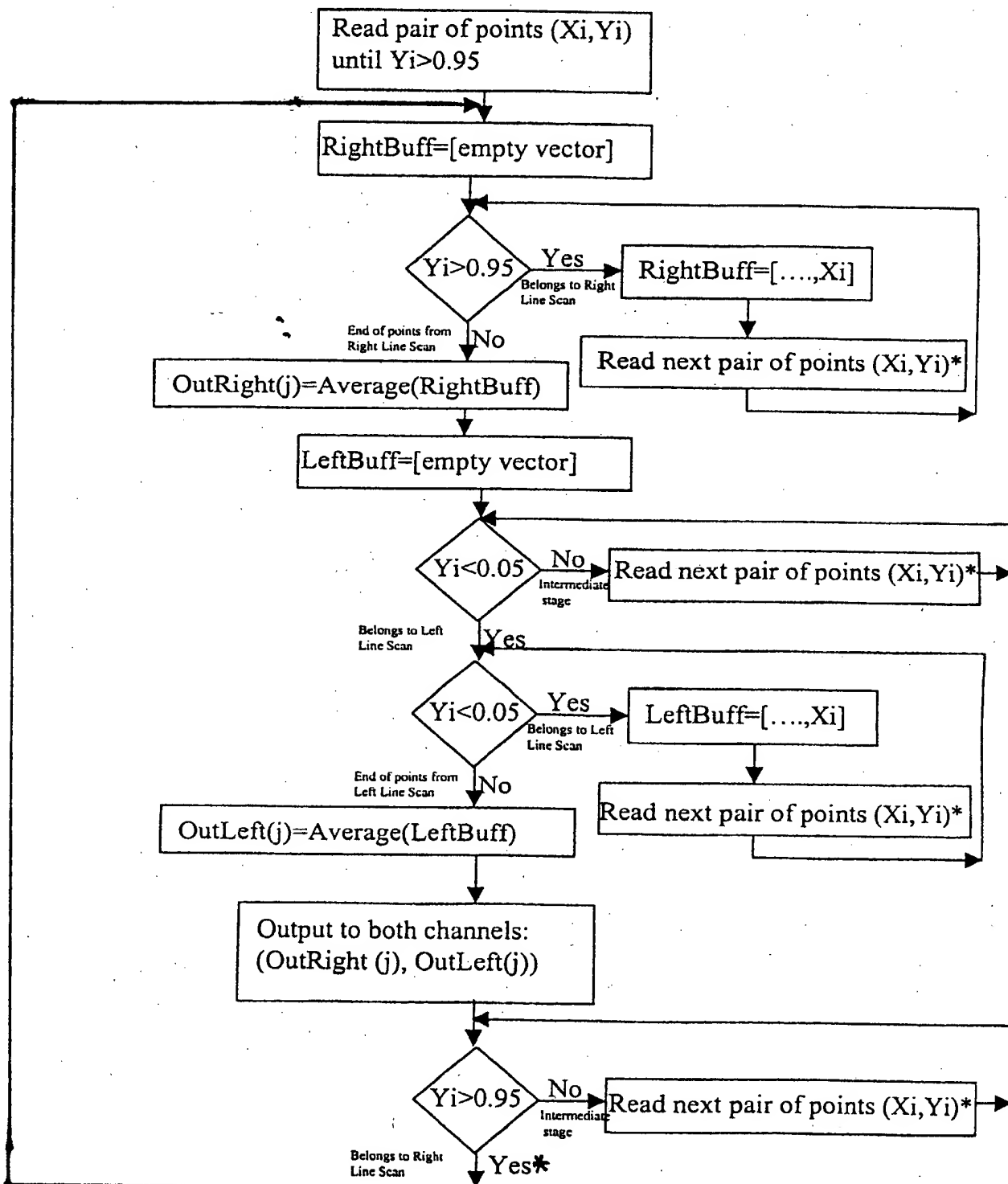
Output - two vectors: OutRight(j) - fluorescence from Right Line scan ⎤ Usually Sampled
                      OutLeft (j) - fluorescence from Left Line scan ⎦ at 5KHz
The sampling rate of the output channels allways equals the frequency of the chopping signal

Read pair of points (Xi,Yi)
until Yi>0.95

RightBuff=[empty vector]

Yi>0.95 → Yes (Belongs to Right Line Scan) → RightBuff=[....,Xi]

End of points from Right Line Scan │No

OutRight(j)=Average(RightBuff)

Read next pair of points (Xi,Yi)*

LeftBuff=[empty vector]

Yi<0.05 → No (Intermediate stage) → Read next pair of points (Xi,Yi)*

Belongs to Left Line Scan │Yes

Yi<0.05 → Yes (Belongs to Left Line Scan) → LeftBuff=[....,Xi]

End of points from Left Line Scan │No

OutLeft(j)=Average(LeftBuff)

Read next pair of points (Xi,Yi)*

Output to both channels:
(OutRight (j), OutLeft(j))

Yi>0.95 → No (Intermediate stage) → Read next pair of points (Xi,Yi)*

Belongs to Right Line Scan │Yes*

* Program ends when input file ends

FIG. 7